STRECPY

The strecpy() and streadd() functions are dangerous unless care is taken to allocate a large enough output buffer.

Sean Barnum, Cigital, Inc. [vita[1]]

2007-04-17

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 4552 bytes

Attack Category	• Malicious Input
Vulnerability Category	• Buffer Overflow
Software Context	• String Management
Location	
Description	The strecpy() and streadd() functions are dangerous unless care is taken to allocate a large enough output buffer. The strecpy(char *theTarget, const char *theSource, const char *exceptions) function is used to copy an input string into a target, expanding non-graphic characters to their escape sequence representations. The string is copied until a null byte is encountered. For example, the compressed version of \t would be expanded out into its escape sequence value. The third argument is a list of characters that are not to be expanded. A pointer to the first byte of the target buffer is returned. This function is a security risk because there is the potential to overflow the target buffer. The risk for this function is greater than that for the functions that compress because a simple test of the size of the source string is not enough to guarantee that the target is large enough.
APIs	<table><tr><th>Function Name</th><th>Comments</th></tr><tr><td>streadd</td><td></td></tr><tr><td>strecpy</td><td></td></tr></table>
Method of Attack	These functions substitute binary characters with string equivalents (i.e. \n, \t, \001). They do not do any bounds checking and are susceptible to buffer overflow by an attacker.
Exception Criteria	

[1] http://buildsecurityin.us-cert.gov/bsi/about_us/authors/35-BSI.html (Barnum, Sean)

<table>
<tr><td><b>Solutions</b></td><td><table>
<tr><th>Solution Applicability</th><th>Solution Description</th><th>Solution Efficacy</th></tr>
<tr><td>Whenever character expanding functions are used.</td><td>Be very cautious using these functions. Check the bounds of the destination buffer to make sure that it is big enough to hold the input after it is expanded. The destination buffer should be AT LEAST four times the size of the input buffer. In the terminal case, a buffer of entire binary characters could have each character replaced with four new characters (e.g.,. \001).</td><td>Effective.</td></tr>
</table></td></tr>
<tr><td><b>Signature Details</b></td><td>char *strecpy (char *output, const char *input, const char *exceptions);
char *streadd (char *output, const char *input, const char *exceptions);</td></tr>
<tr><td><b>Examples of Incorrect Code</b></td><td>char theSource[numberOfCharactersInTheSource]="st
\t\tlike these\n\n";
char theTarget[numberOfCharactersInTheSource];
strecpy(theTarget,theSource,theExceptions);

/* In this case, if the characters are expanded, then the target buffer will overflow. */</td></tr>
<tr><td><b>Examples of Corrected Code</b></td><td>char theSource[numberOfCharactersInTheSource]="st
\t\tlike these\n\n";
char theTarget[4*numberOfCharactersInTheSource];
strecpy(theTarget,theSource,theExceptions);</td></tr>
</table>

<table>
<tr><td></td><td>/* In this case, then the target buffer will not overflow. */</td></tr>
<tr><td><b>Source Reference</b></td><td>• Viega, John & McGraw, Gary. <i>Building Secure Software: How to Avoid Security Problems the Right Way</i>. Boston, MA: Addison-Wesley Professional, 2001, ISBN: 020172152X, p. 146.</td></tr>
<tr><td><b>Recommended Resource</b></td><td></td></tr>
<tr><td><b>Discriminant Set</b></td><td><table><tr><td><b>Operating System</b></td><td>• UNIX</td></tr><tr><td><b>Languages</b></td><td>• C
• C++</td></tr></table></td></tr>
</table>

Cigital, Inc. Copyright

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

[1] mailto:copyright@cigital.com